Analysis of Unicorn Startups

Contents

1 Setup

1.1 Import Packages

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter
import seaborn as sns
```

1.2 Theming

```
sns.set_theme(palette='husl')
```

2 Introduction

- **What is a Unicorn Startup?**

  In business, a unicorn is a startup company valued at over US$1 billion which is privately owned and not listed on a share market.

3 Data Preparation

3.1 Load Data

```
pd.set_option('display.max_columns', 50, 'display.width', 200)
df = pd.read_csv('input/datasets/Unicorns_Completed (2024).csv')
df_latest = pd.read_csv('input/raw_data/list-of-unicorn-startups_20250619 (wikipedia).csv')
```

3.2 Data Cleaning

```
import re
def convert_years_months(s):
    m = re.match(r'(\d+)y?\s?(\d+)m?o?', s)
    return f'{m[1]}y{m[2]}m' if m else s

df['Years to Unicorn'] = df['Years to Unicorn'].apply(convert_years_months)

def correct_industry_labels(s):
    if s == 'Health':
        return 'Healthcare & Life Sciences'
    if s == 'West Palm Beach':
        return 'Enterprise Tech'
    return s

df['Industry'] = df['Industry'].apply(correct_industry_labels)

def correct_company_names(s):
    if s == 'Scale':
        return 'Scale AI'
    return s

df['Company'] = df['Company'].apply(correct_company_names)
```

3.3 Prepare Data

```
df['Unicorn Date'] = pd.to_datetime(df['Unicorn Date'])
df['Valuation ($B)'] = pd.to_numeric(df['Valuation ($B)'])
df['Unicorn Year'] = df['Unicorn Date'].dt.year
df['Funding ($B)'] = df['Total Equity Funding ($)'] / 1e9

df_latest.rename(columns={'Valuation (US$ billions)': 'Latest Valuation ($B)'},
↪  inplace=True)
df_latest = df_latest.drop_duplicates('Company')
df_latest['Company'] = df_latest['Company'].str.strip()
df = df.merge(df_latest[['Company', 'Latest Valuation ($B)']], on='Company', how='left')
df['Latest Valuation ($B)'] = pd.to_numeric(df['Latest Valuation
↪  ($B)'].fillna(value=df['Valuation ($B)']))
```

3.4 Preview

```
print(df.info())
print(df.describe())
df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 1244 entries, 0 to 1243
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Company                   1244 non-null   object
 1   Valuation ($B)            1244 non-null   float64
 2   Total Equity Funding ($)  1244 non-null   int64
 3   Unicorn Date              1244 non-null   datetime64[ns]
 4   Date Founded              1244 non-null   int64
 5   Years to Unicorn          1244 non-null   object
 6   Industry                  1244 non-null   object
 7   Country                   1244 non-null   object
 8   City                      1244 non-null   object
 9   Select Investors          1244 non-null   object
 10  Unicorn Year              1244 non-null   int32
 11  Funding ($B)              1244 non-null   float64
 12  Latest Valuation ($B)     1244 non-null   float64
dtypes: datetime64[ns](1), float64(3), int32(1), int64(2), object(6)
memory usage: 121.6+ KB
None
```

	Valuation ($B)	Total Equity Funding ($)	Unicorn Date	Date Fou
count	1244.000000	1.244000e+03	1244	1244.00
mean	3.626487	5.985096e+08	2021-02-10 22:05:24.115755776	2013.37
min	1.000000	0.000000e+00	2007-07-02 00:00:00	1919.00
25%	1.100000	2.170000e+08	2020-08-13 12:00:00	2011.00
50%	1.550000	3.525000e+08	2021-07-21 00:00:00	2014.00
75%	3.000000	6.090000e+08	2022-02-24 00:00:00	2017.00
max	350.000000	1.900000e+10	2024-12-24 00:00:00	2024.00
std	15.016365	1.222045e+09	NaN	5.51

4 Descriptive Analysis

4.1 Distribution

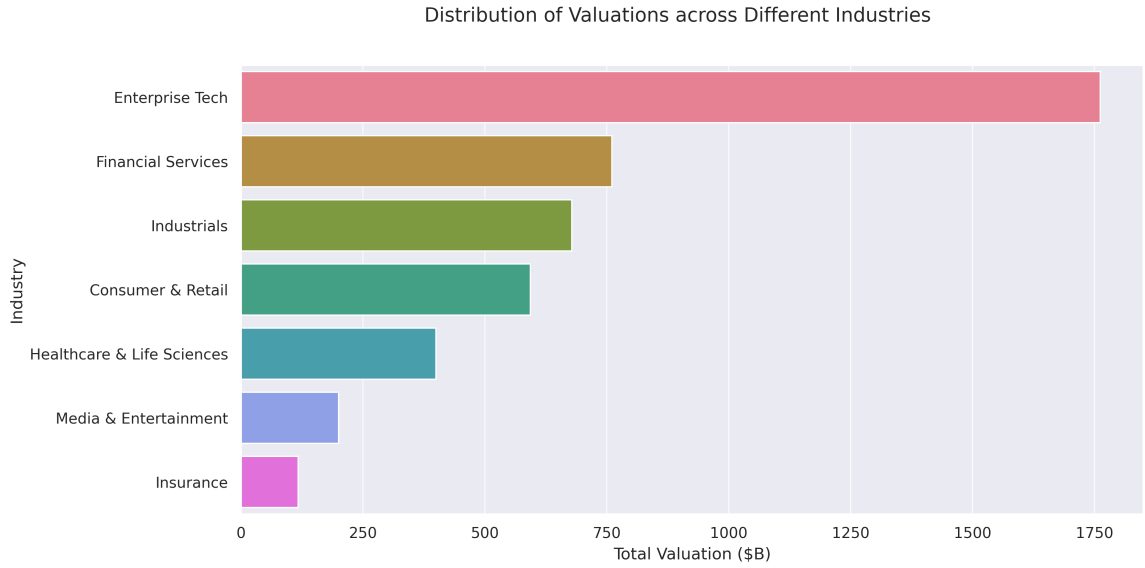
4.1.1 Valuations

Distribution of Valuations across Different Industries

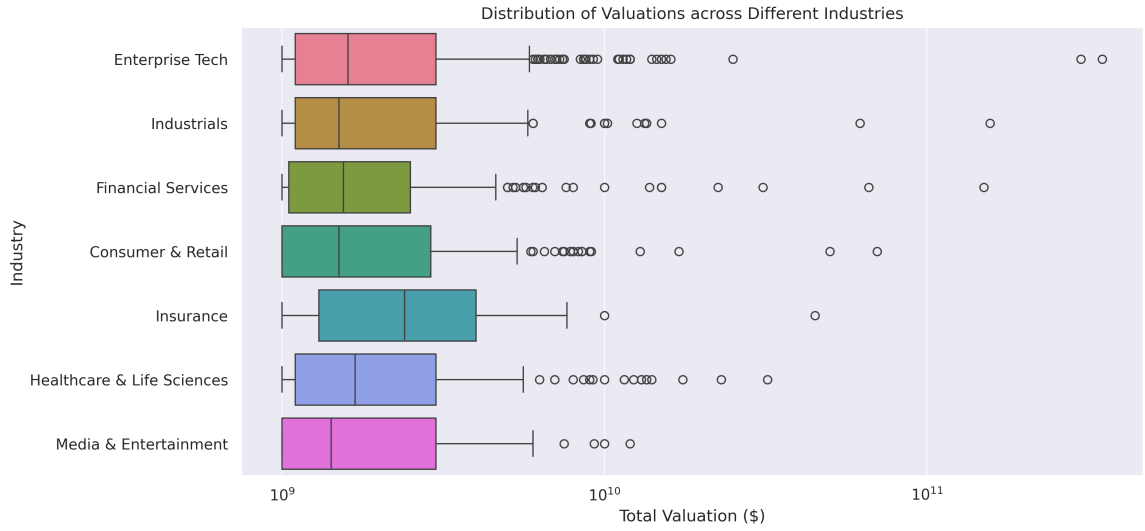
```
# Group by industry and sum valuations
industry_valuation_df = df.groupby('Industry')['Valuation
↪  ($B)'].sum().reset_index().sort_values('Valuation ($B)', ascending=False)
industry_valuation_df
```

```
plt.subplots(figsize=(12, 6), dpi=300)
```

```
ax = sns.barplot(industry_valuation_df,
                 y='Industry',
                 x='Valuation ($B)',
                 hue='Industry')
plt.xlabel('Total Valuation ($B)')
plt.ylabel('Industry')
plt.suptitle('Distribution of Valuations across Different Industries')
plt.grid(axis='x', alpha=0.75)
plt.show()
```



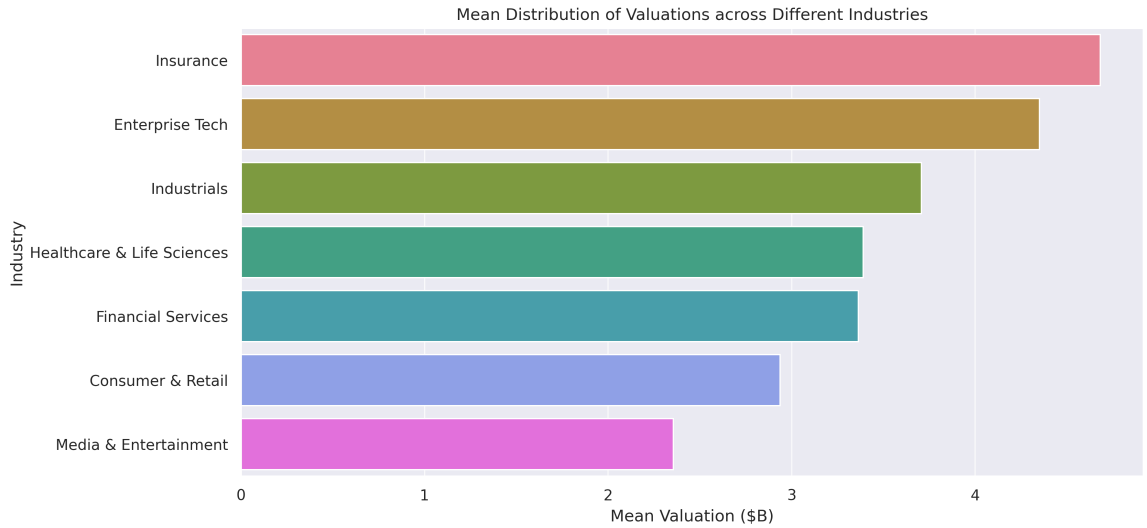

```
plt.subplots(figsize=(12, 6), dpi=300)
sns.boxplot(df, y='Industry', x=df['Valuation ($B)']*1e9, hue='Industry')
plt.title('Distribution of Valuations across Different Industries')
plt.xlabel('Total Valuation ($)')
plt.ylabel('Industry')
plt.xscale('log')
plt.grid(axis='x', alpha=0.7)
plt.show()
```



Mean Distribution of Valuations across Different Industries

```
industry_valuation_df = df.groupby('Industry')['Valuation
↪  ($B)'].mean().reset_index().sort_values('Valuation ($B)', ascending=False)
industry_valuation_df
```

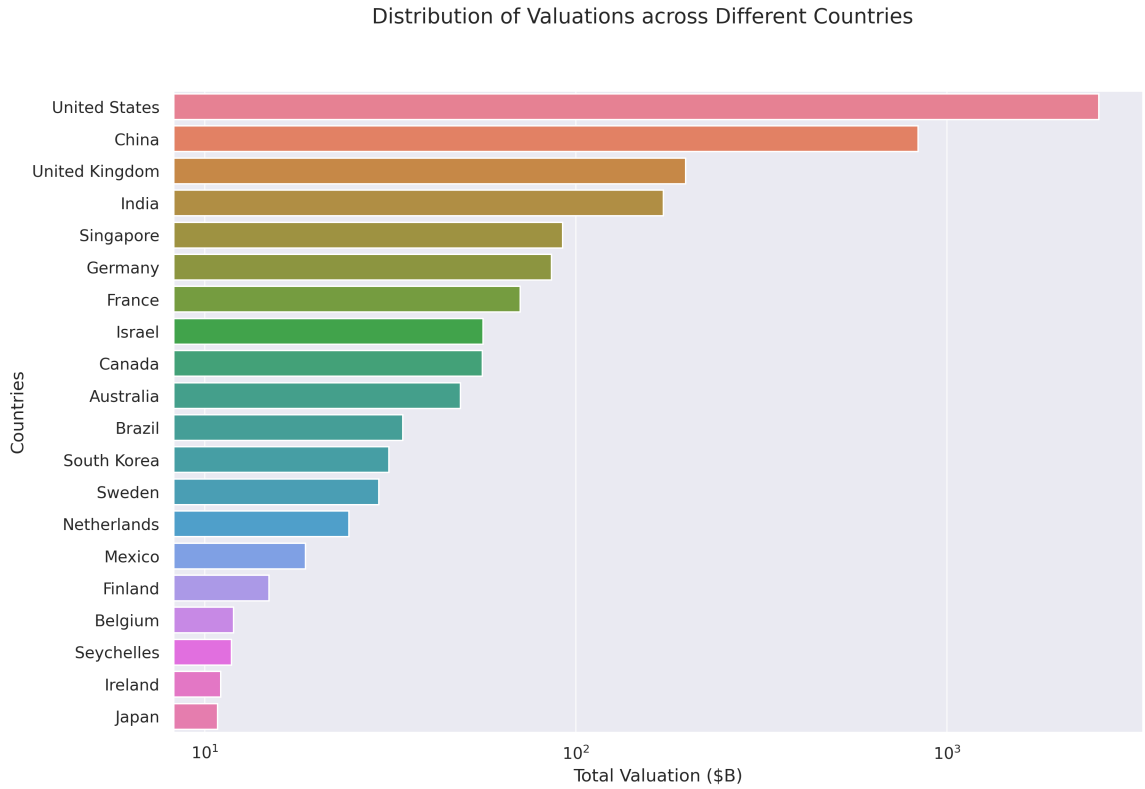
```
plt.figure(figsize=(12, 6),dpi=300)
sns.barplot(industry_valuation_df,
            y='Industry',
            x='Valuation ($B)',
            hue='Industry')
plt.title('Mean Distribution of Valuations across Different Industries')
plt.xlabel('Mean Valuation ($B)')
plt.ylabel('Industry')
plt.grid(axis='x', alpha=0.75)
```



Distribution of Valuations across Different Countries

```
# Group by Country and sum valuations
country_valuation_df = df.groupby('Country')['Valuation
↪  ($B)'].sum().reset_index().sort_values('Valuation ($B)', ascending=False).head(20)
country_valuation_df
```

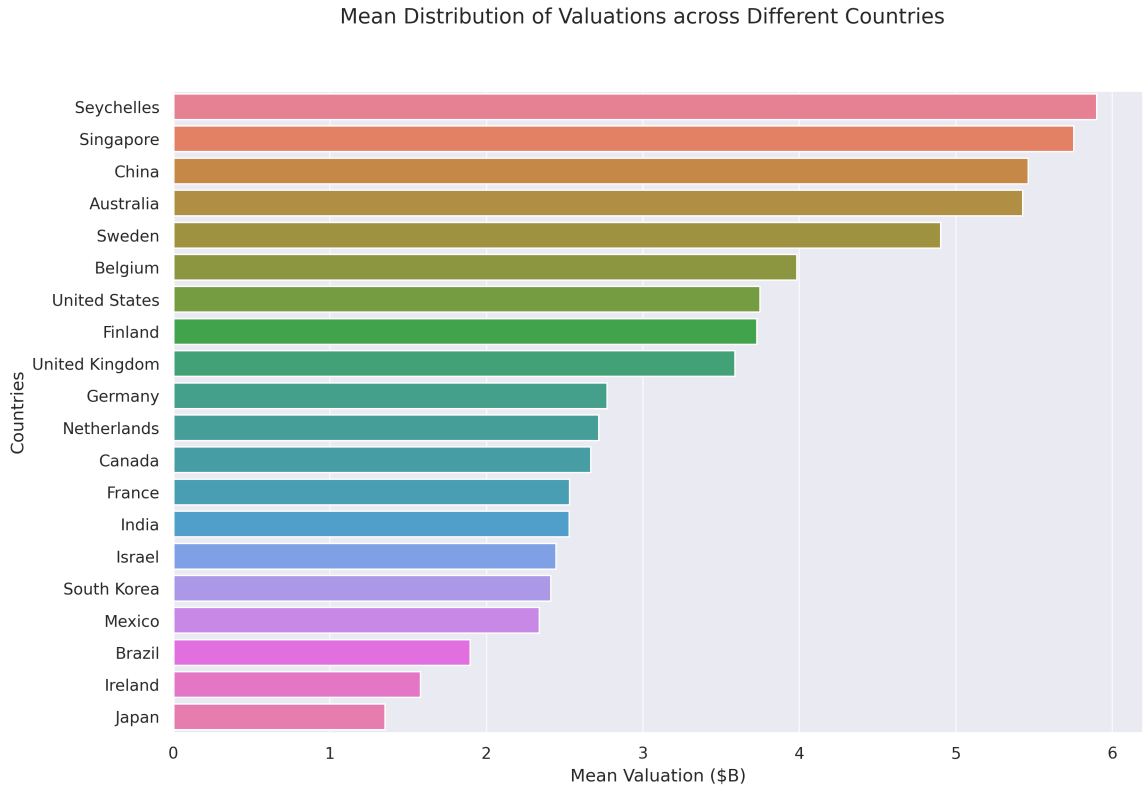
```
plt.subplots(figsize=(12, 8), dpi=300)
sns.barplot(country_valuation_df,
            y='Country',
            x='Valuation ($B)',
            hue='Country')
plt.suptitle('Distribution of Valuations across Different Countries')
plt.xlabel('Total Valuation ($B)')
plt.ylabel('Countries')
plt.grid(axis='x', alpha=0.75)
plt.xscale('log')
plt.show()
```



Mean Distribution of Valuations across Different Countries

```
mean_country_valuation_df =
↪  df[df['Country'].isin(country_valuation_df['Country'])].groupby('Country')['Valuation
↪  ($B)'].mean().reset_index().sort_values('Valuation ($B)', ascending=False).head(20)
mean_country_valuation_df
```

```
plt.figure(figsize=(12, 8), dpi=300)
sns.barplot(mean_country_valuation_df,
            y='Country',
            x='Valuation ($B)',
            hue='Country')
plt.suptitle('Mean Distribution of Valuations across Different Countries')
plt.xlabel('Mean Valuation ($B)')
plt.ylabel('Countries')
plt.grid(axis='x', alpha=0.75)
plt.show()
```

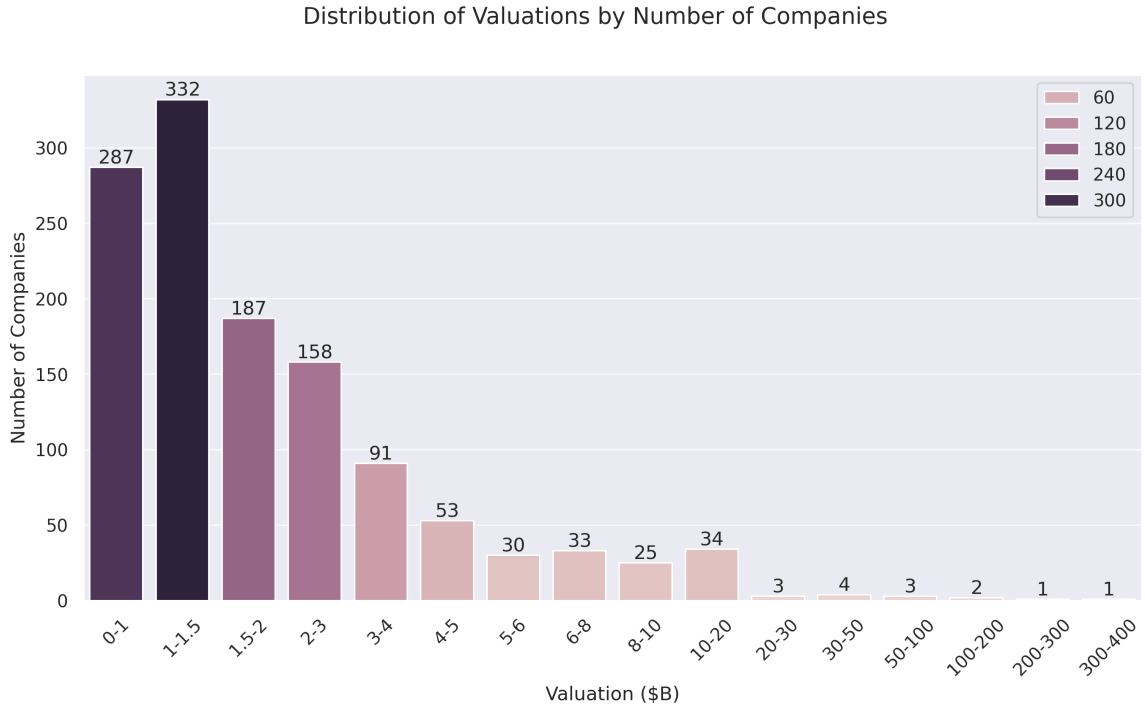


Distribution of Valuations by Number of Companies

```
# Define the bins for valuation ranges
bins = [0, 1, 1.5, 2, 3, 4, 5, 6, 8, 10, 20, 30, 50, 100, 200, 300, 400]
labels =  [f'{a}-{b}' for a, b in zip(bins[:-1], bins[1:])]
cuts = pd.cut(df['Valuation ($B)'], bins=bins, labels=labels)

# Count the number of companies in each bin
valuation_distribution = cuts.value_counts().sort_index()

# Plot the Bar Chart
plt.figure(figsize=(12, 6), dpi=300)
ax = sns.barplot(x=valuation_distribution.index,
                 y=valuation_distribution.values, hue=valuation_distribution.values)
for i in ax.containers:
    ax.bar_label(i)
plt.suptitle('Distribution of Valuations by Number of Companies')
plt.xlabel('Valuation ($B)')
plt.ylabel('Number of Companies')
plt.xticks(rotation=45)
plt.grid(axis='y', alpha=0.75)
#plt.yscale('log')
plt.show()
```

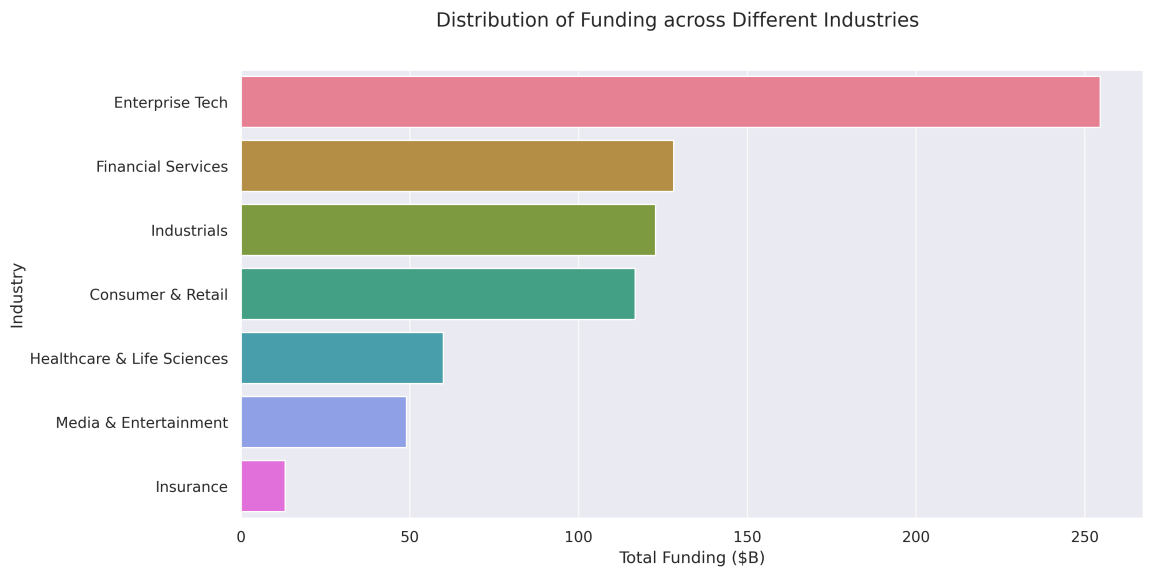


4.1.2 Funding

Distribution of Funding across Different Industries

```
# Group by industry and sum valuations
industry_funding_df = df.groupby('Industry')['Funding
↪  ($B)'].sum().reset_index().sort_values('Funding ($B)', ascending=False)
industry_funding_df
```

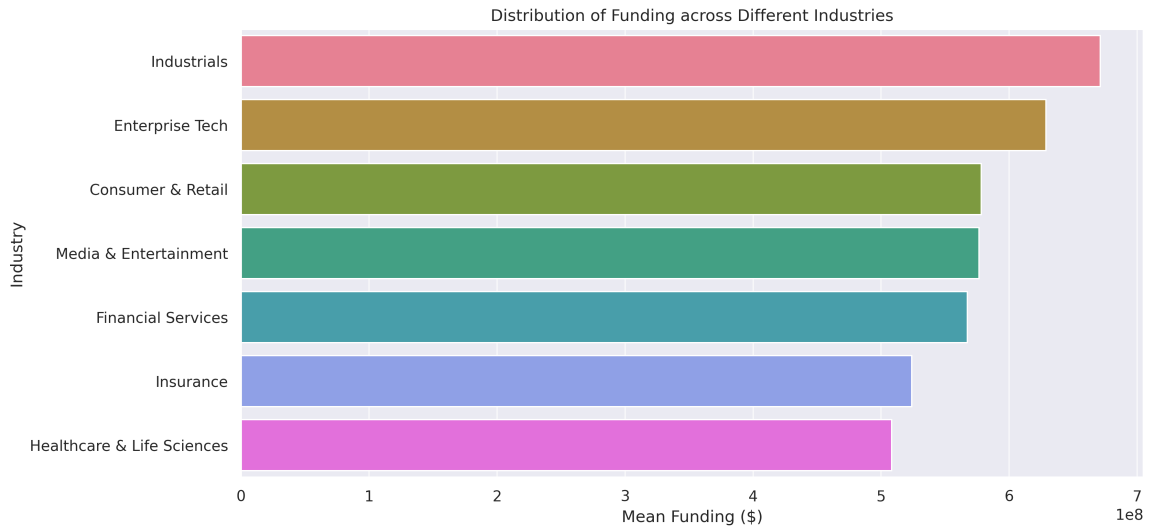
```
plt.figure(figsize=(12, 6), dpi=300)
sns.barplot(industry_funding_df,
            y='Industry', x='Funding ($B)', hue='Industry')
plt.suptitle('Distribution of Funding across Different Industries')
plt.xlabel('Total Funding ($B)')
plt.ylabel('Industry')
plt.grid(axis='x', alpha=0.75)
```



Mean Distribution of Funding across Different Industries

```
industry_funding_df = df.groupby('Industry')['Total Equity Funding
↪  ($)'].mean().reset_index().sort_values('Total Equity Funding ($)', ascending=False)
industry_funding_df
```

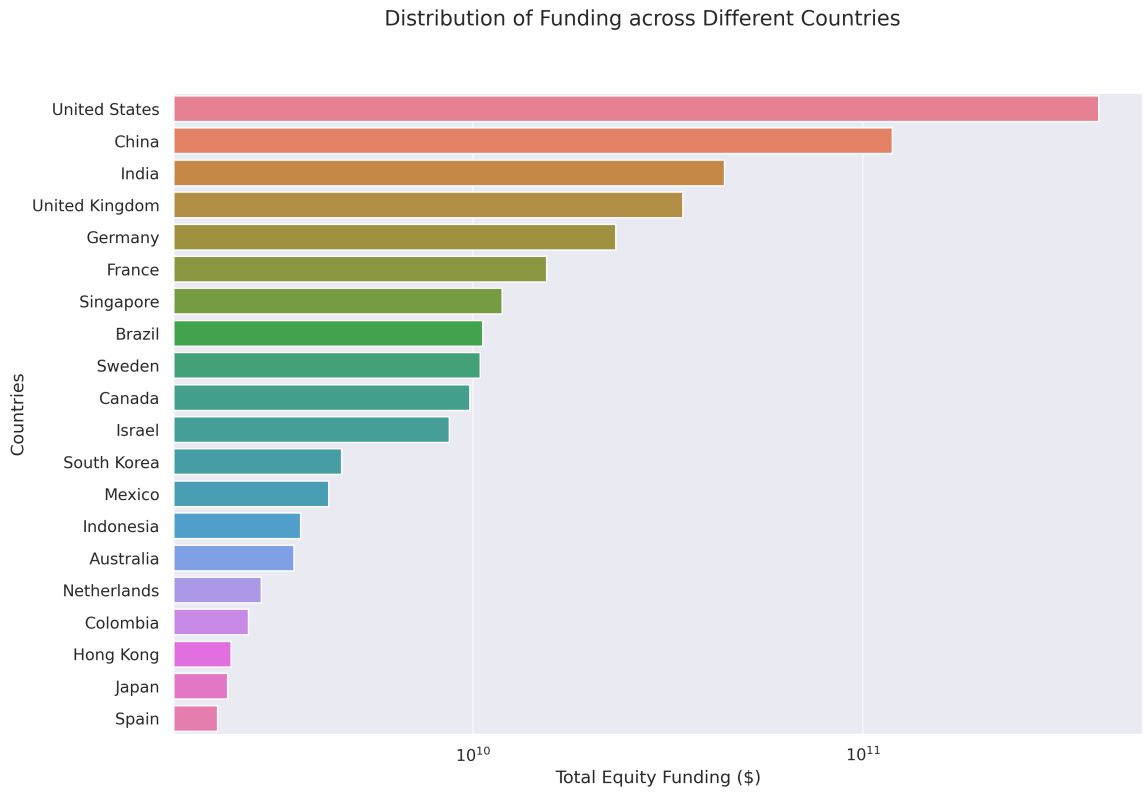
```
plt.figure(figsize=(12, 6), dpi=300)
sns.barplot(industry_funding_df,
            y='Industry',
            x='Total Equity Funding ($)',
            hue='Industry')
plt.title('Distribution of Funding across Different Industries')
plt.xlabel('Mean Funding ($)')
plt.ylabel('Industry')
plt.grid(axis='x', alpha=0.75)
plt.show()
```



Distribution of Funding across Different Countries

```
# Group by Country and sum valuations
country_funding_df = df.groupby('Country')['Total Equity Funding
↪  ($)'].sum().reset_index().sort_values('Total Equity Funding ($)',
↪  ascending=False).head(20)
country_funding_df
```

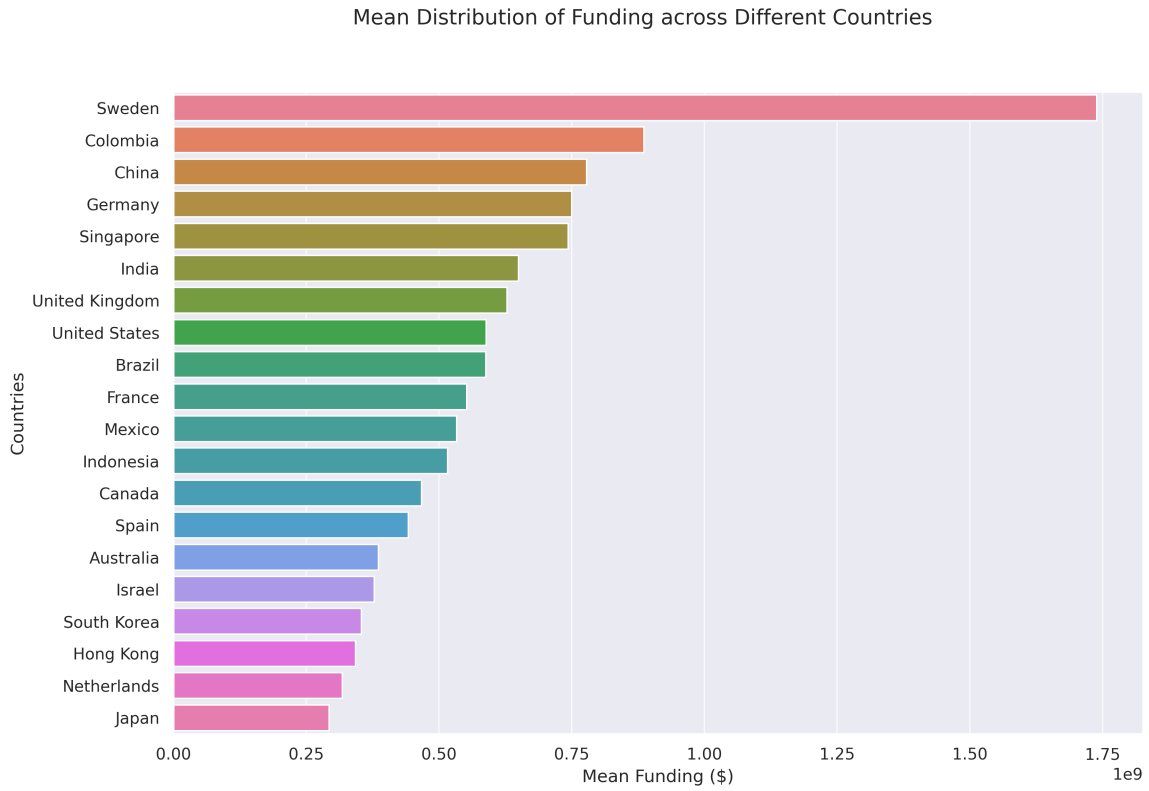
```
plt.figure(figsize=(12, 8), dpi=300)
sns.barplot(country_funding_df, y='Country', x='Total Equity Funding ($)', hue='Country')
plt.suptitle('Distribution of Funding across Different Countries')
plt.xlabel('Total Equity Funding ($)')
plt.ylabel('Countries')
plt.grid(axis='x', alpha=0.75)
plt.xscale('log')
plt.show()
```



Mean Distribution of Funding across Different Countries

```
# Group by Country and sum valuations
mean_country_funding_df =
↪  df[df['Country'].isin(country_funding_df['Country'])].groupby('Country')['Total Equity
↪  Funding ($)'].mean().reset_index().sort_values('Total Equity Funding ($)',
↪  ascending=False).head(20)
mean_country_funding_df
```

```
plt.figure(figsize=(12, 8), dpi=300)
sns.barplot(mean_country_funding_df,
            y='Country',
            x='Total Equity Funding ($)',
            hue='Country')
plt.suptitle('Mean Distribution of Funding across Different Countries')
plt.xlabel('Mean Funding ($)')
plt.ylabel('Countries')
plt.grid(axis='x', alpha=0.75)
plt.show()
```



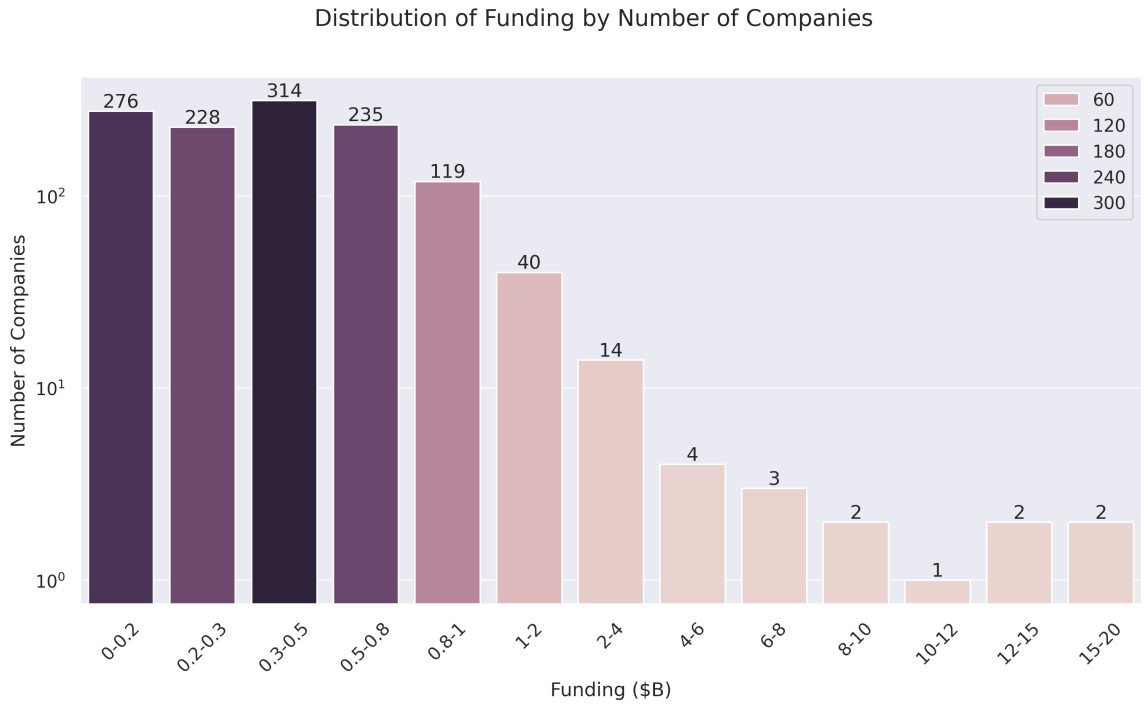
Distribution of Funding by Number of Companies

```
# Define the bins for funding ranges
bins = [0, 0.2, 0.3, 0.5, 0.8, 1, 2, 4, 6, 8, 10, 12, 15, 20]
labels =  [f'{a}-{b}' for a, b in zip(bins[:-1], bins[1:])]
cuts = pd.cut(df['Funding ($B)'], bins=bins, labels=labels)

# Count the number of companies in each bin
funding_distribution = cuts.value_counts().sort_index()

# Plot the Bar Chart
plt.figure(figsize=(12, 6), dpi=300)
ax = sns.barplot(x=funding_distribution.index,
                 y=funding_distribution.values, hue=funding_distribution.values)
for i in ax.containers:
    ax.bar_label(i)
```

```
plt.suptitle('Distribution of Funding by Number of Companies')
plt.xlabel('Funding ($B)')
plt.ylabel('Number of Companies')
plt.xticks(rotation=45)
plt.grid(axis='y', alpha=0.75)
plt.yscale('log')
plt.show()
```



5 Comparative Analysis

5.1 By Company

5.1.1 Top Companies by Valuation

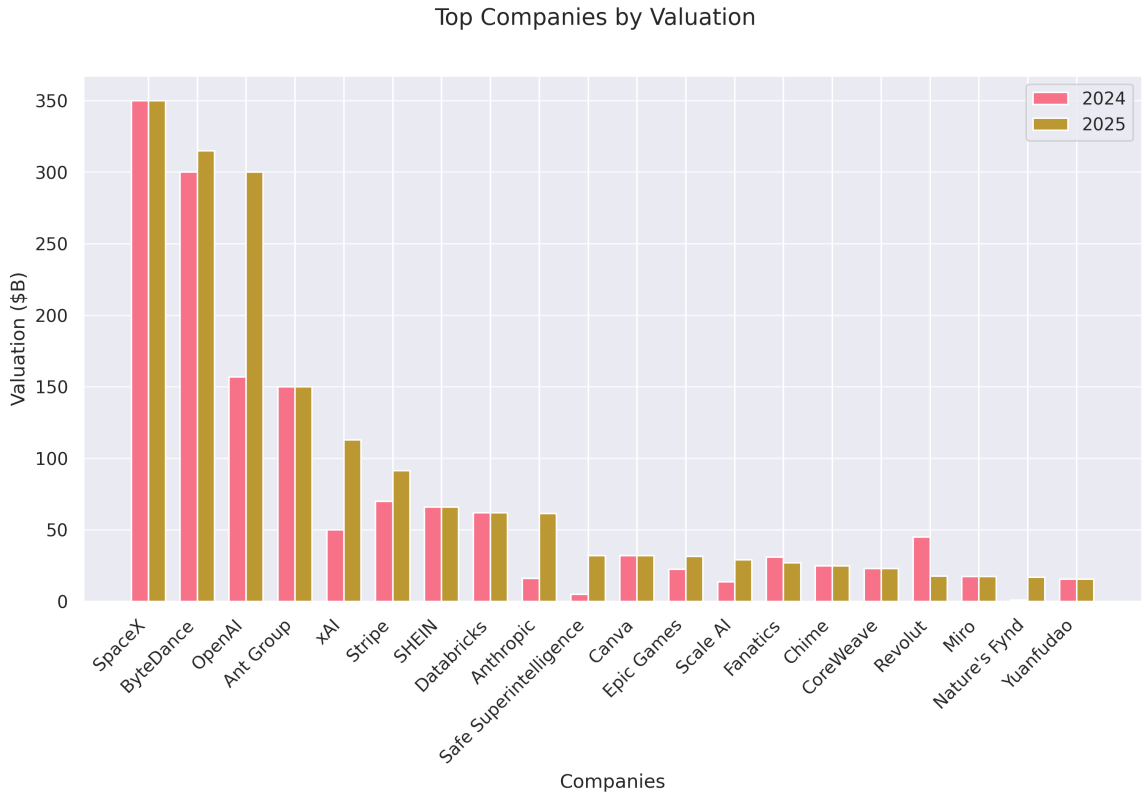
```
top_companies = df.sort_values(by='Latest Valuation ($B)', ascending=False).head(20)
top_companies
```

```
# Set the positions and width for the bars
N = len(top_companies)
ind = np.arange(N)  # the x locations for the groups
width = 0.35  # the width of the bars

# Create the bars for valuation and funding
fig, ax = plt.subplots(figsize=(12, 6), dpi=300)
bars1 = ax.bar(ind, top_companies['Valuation ($B)'], width, label='2024')
bars2 = ax.bar(ind + width, top_companies['Latest Valuation ($B)'], width, label='2025')

# Add labels and title
```

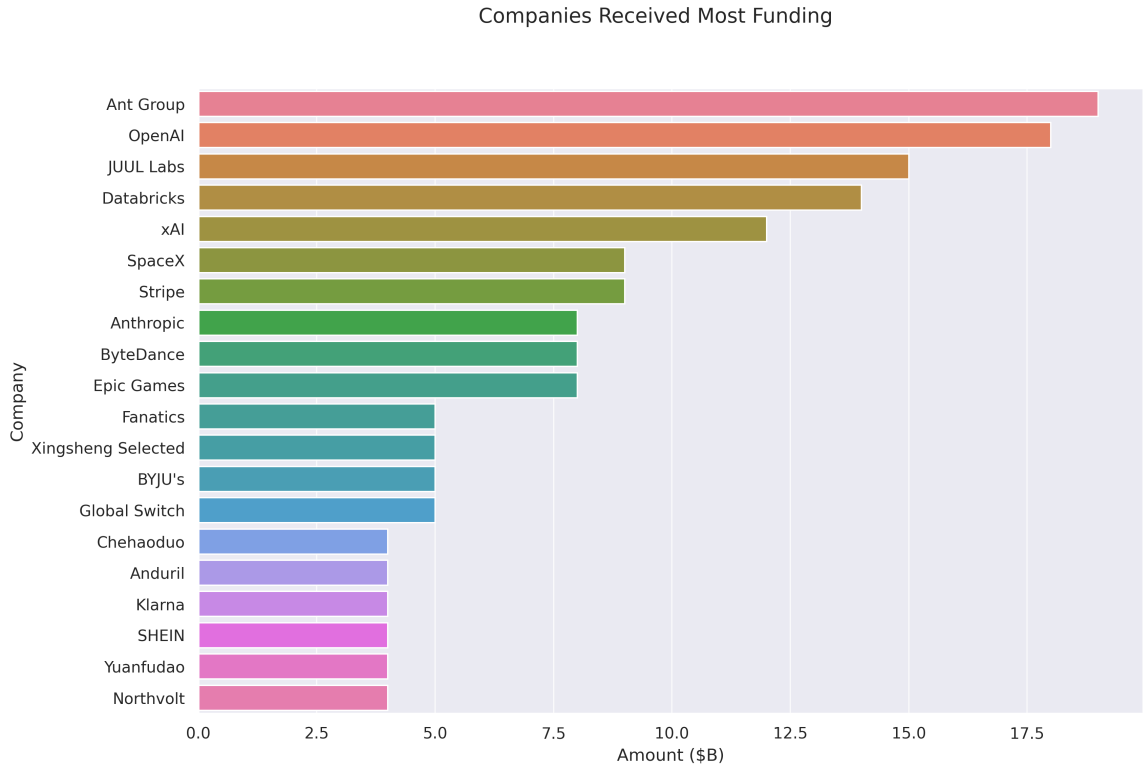
```
ax.set(xlabel='Companies',
       ylabel='Valuation ($B)')
ax.set_xticks(ind+width/2, top_companies['Company'], rotation=45, ha='right')
ax.legend()
ax.grid(axis='y', alpha=0.75)
plt.suptitle('Top Companies by Valuation')
plt.show()
```



5.1.2 Companies Received Most Funding

```
top_companies = df.sort_values(by='Funding ($B)', ascending=False).head(20)
top_companies
```

```
plt.subplots(figsize=(12, 8), dpi=300)
sns.barplot(top_companies, y='Company', x='Funding ($B)', hue='Company')
plt.suptitle('Companies Received Most Funding')
plt.xlabel('Amount ($B)')
plt.grid(axis='x', alpha=0.75)
plt.show()
```



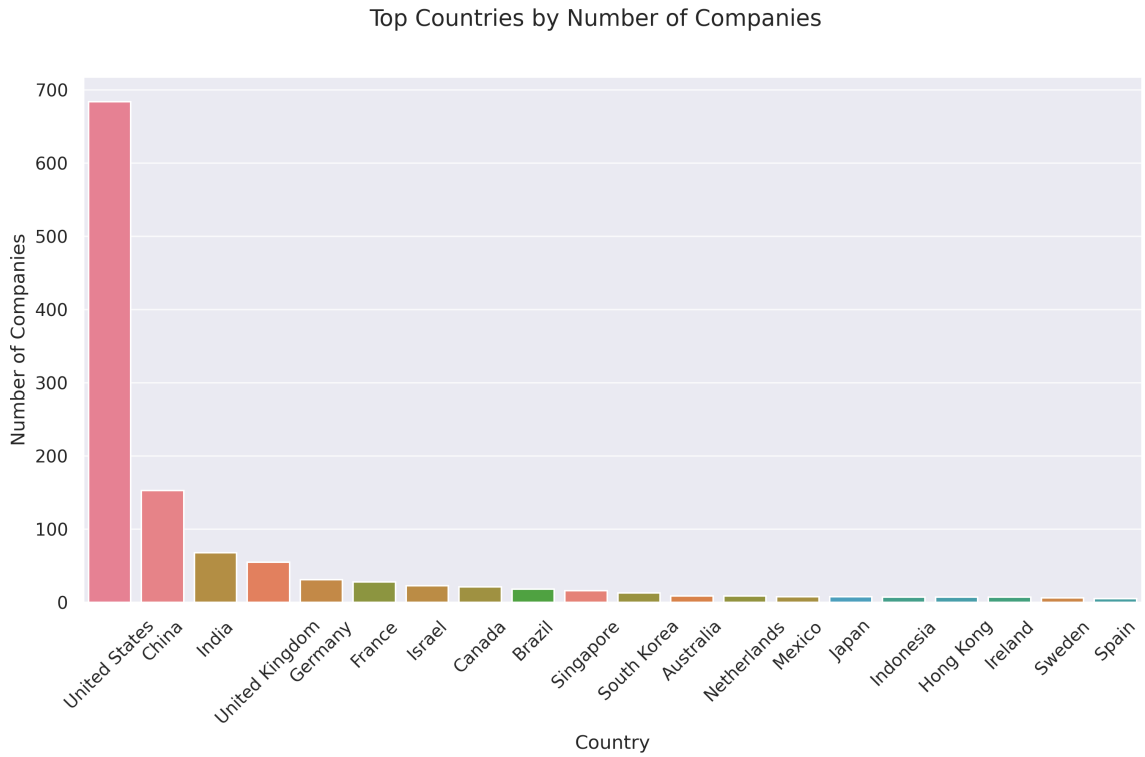
5.2 By Country

```
top_countries = df['Country'].value_counts().nlargest(5).index
top_countries
```

```
Index(['United States', 'China', 'India', 'United Kingdom', 'Germany'], dtype='object',
```

5.2.1 Top Countries by Number of Companies

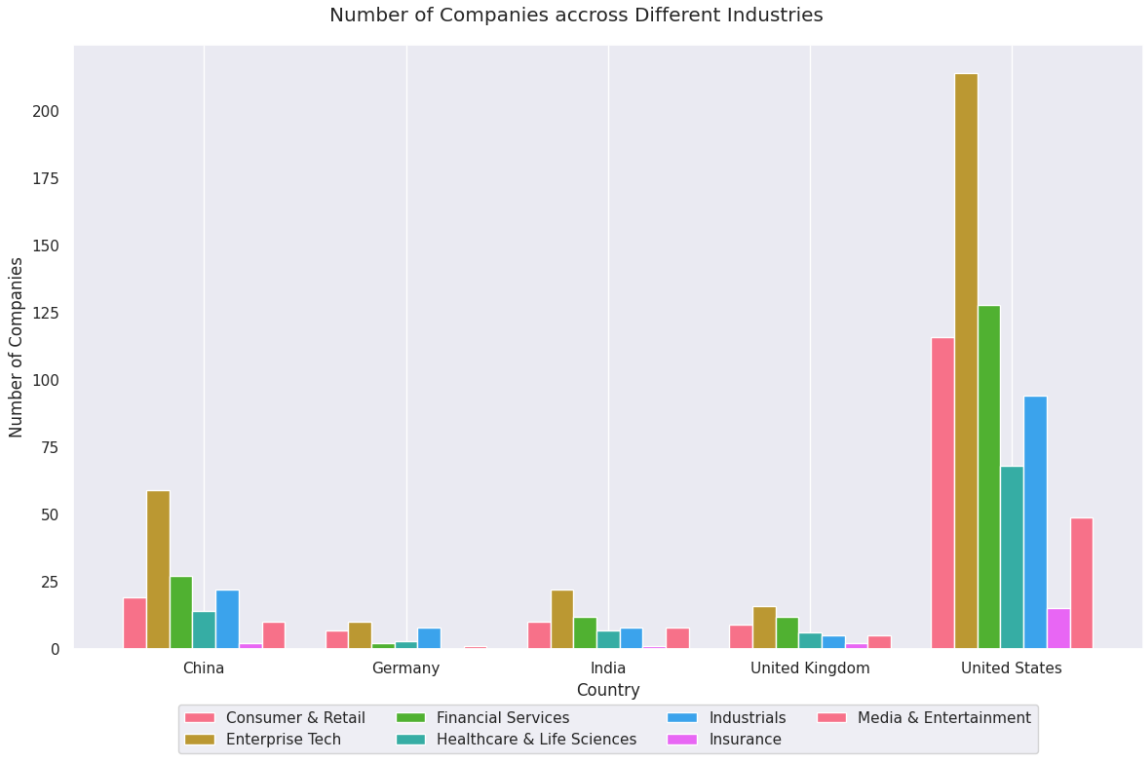
```
plt.subplots(figsize=(12, 6), dpi=300)
sns.countplot(x=df['Country'],
              order=df['Country'].value_counts().nlargest(20).index,
              hue=df['Country'])
plt.suptitle('Top Countries by Number of Companies')
plt.ylabel('Number of Companies')
plt.xticks(rotation=45)
plt.grid(axis='y', alpha=0.75)
plt.show()
```



5.2.2 Top Countries by Number of Companies across Different Industries

```
grouped_df = df[df['Country'].isin(top_countries)].groupby(['Country',
↪  'Industry']).size().unstack(fill_value=0)
grouped_df
```

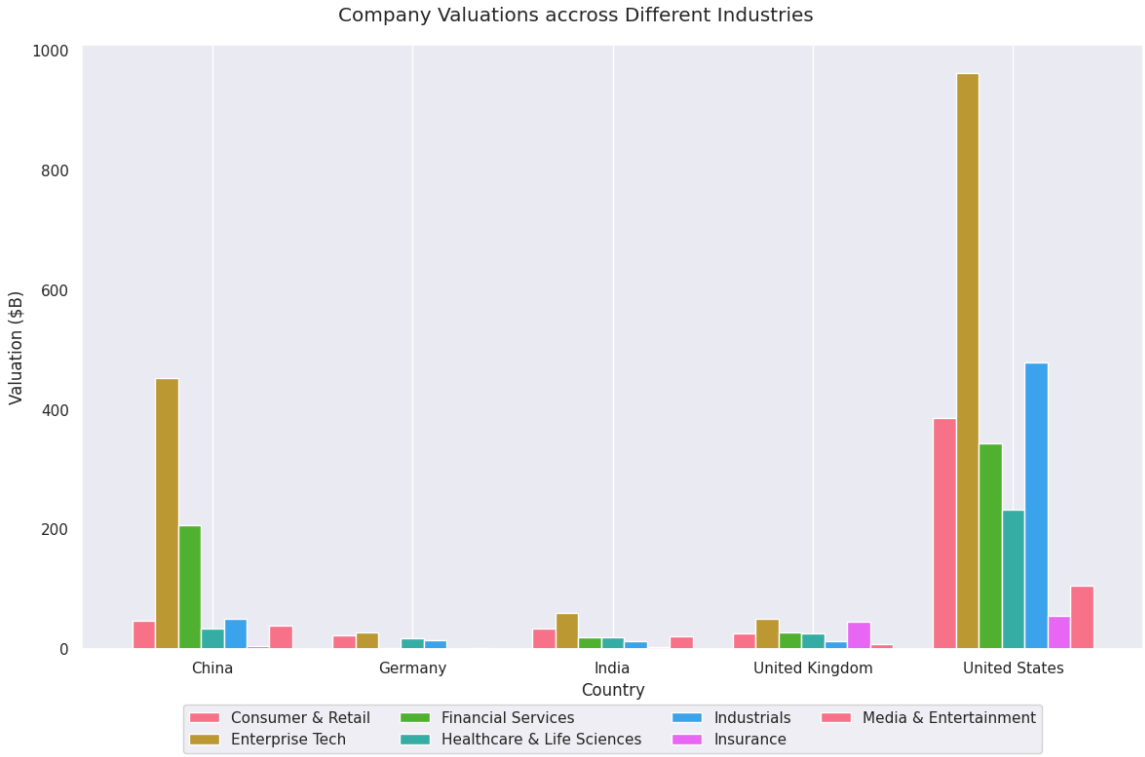
```
grouped_df.plot(kind='bar', figsize=(12, 8), width=0.8)
plt.suptitle('Number of Companies accross Different Industries')
plt.xlabel('Country')
plt.ylabel('Number of Companies')
plt.xticks(rotation=0)  # Keep x-axis labels horizontal
plt.legend(ncol=4, loc="upper center", bbox_to_anchor=(0.5,-0.08))
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```



5.2.3 Top Countries by Company Valuations across Different Industries

```
grouped_df = df[df['Country'].isin(top_countries)].groupby(['Country',
↪  'Industry'])['Valuation ($B)'].sum().unstack(fill_value=0)
grouped_df
```

```
grouped_df.plot(kind='bar', figsize=(12, 8), width=0.8)
plt.suptitle('Company Valuations accross Different Industries')
plt.xlabel('Country')
plt.ylabel('Valuation ($B)')
plt.xticks(rotation=0)  # Keep x-axis labels horizontal
plt.legend(ncol=4, loc="upper center", bbox_to_anchor=(0.5,-0.08))
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

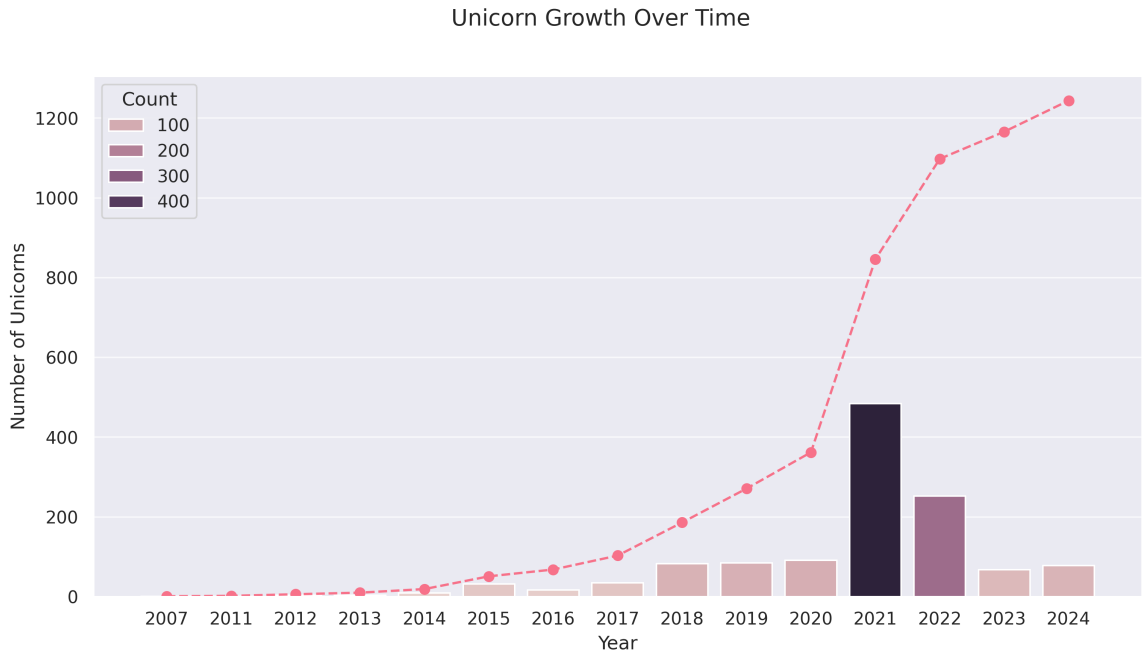


6 Time-Based Analysis

6.1 Unicorn Growth Over Time

```
grouped_df = df.groupby('Unicorn Year').size().reset_index(name='Count')
grouped_df['Accumulated Count'] = grouped_df['Count'].cumsum()
grouped_df
```

```
plt.subplots(figsize=(12, 6), dpi=300)
sns.barplot(grouped_df, x='Unicorn Year', y='Count', hue='Count')
plt.plot(grouped_df['Accumulated Count'], marker='o', linestyle='dashed')
plt.suptitle('Unicorn Growth Over Time')
plt.xlabel('Year')
plt.ylabel('Number of Unicorns')
plt.grid(axis='y', alpha=0.7)
plt.show()
```



The surge of unicorns was reported as "meteoric" for 2021, with $71 billion invested in 340 new companies, a banner year for startups and for the US venture capital industry; the unprecedented number of companies valued at more than $1 billion during 2021 exceeded the sum total of the five previous years.

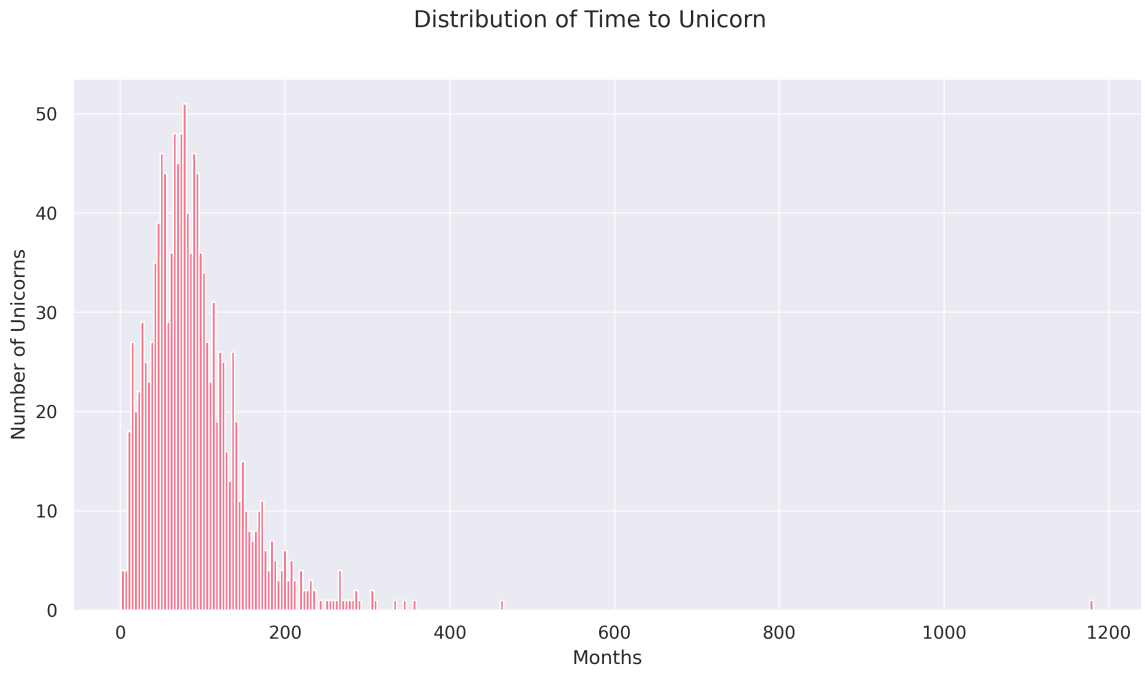
6.2 Time to Unicorn

```
# Function to convert "Years to Unicorn" into total months
def convert_years_to_months(years_str):
    if 'y' in years_str and 'm' in years_str:
        years, months = years_str.split('y')
        months = months.replace('m', '').strip()
        return int(years.strip()) * 12 + int(months)
    elif 'y' in years_str:
        years = years_str.replace('y', '').strip()
        return int(years) * 12
    elif 'm' in years_str:
        months = years_str.replace('mo', '').replace('m', '').strip()
        return int(months)
    else:
        return None

df['Years to Unicorn (Months)'] = df['Years to Unicorn'].apply(convert_years_to_months)
```

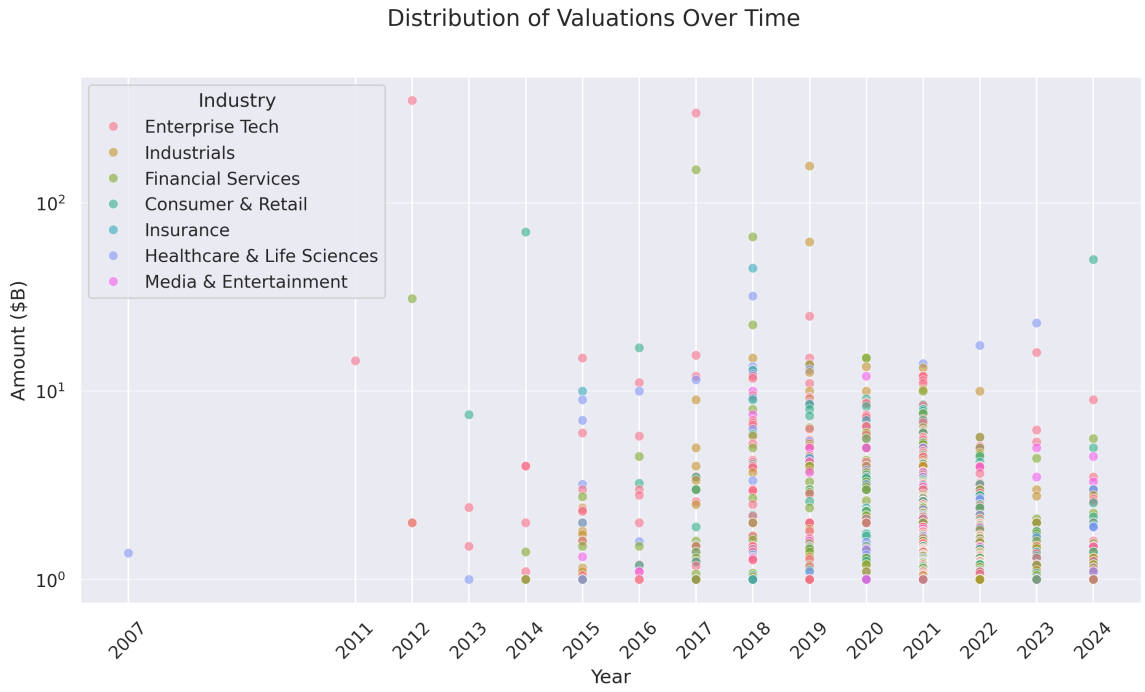
```
plt.subplots(figsize=(12, 6), dpi=300)
plt.hist(df['Years to Unicorn (Months)'].dropna(), bins=300)
plt.suptitle('Distribution of Time to Unicorn')
plt.xlabel('Months')
plt.ylabel('Number of Unicorns')
plt.grid(alpha=0.75)
```

```
plt.show()
```



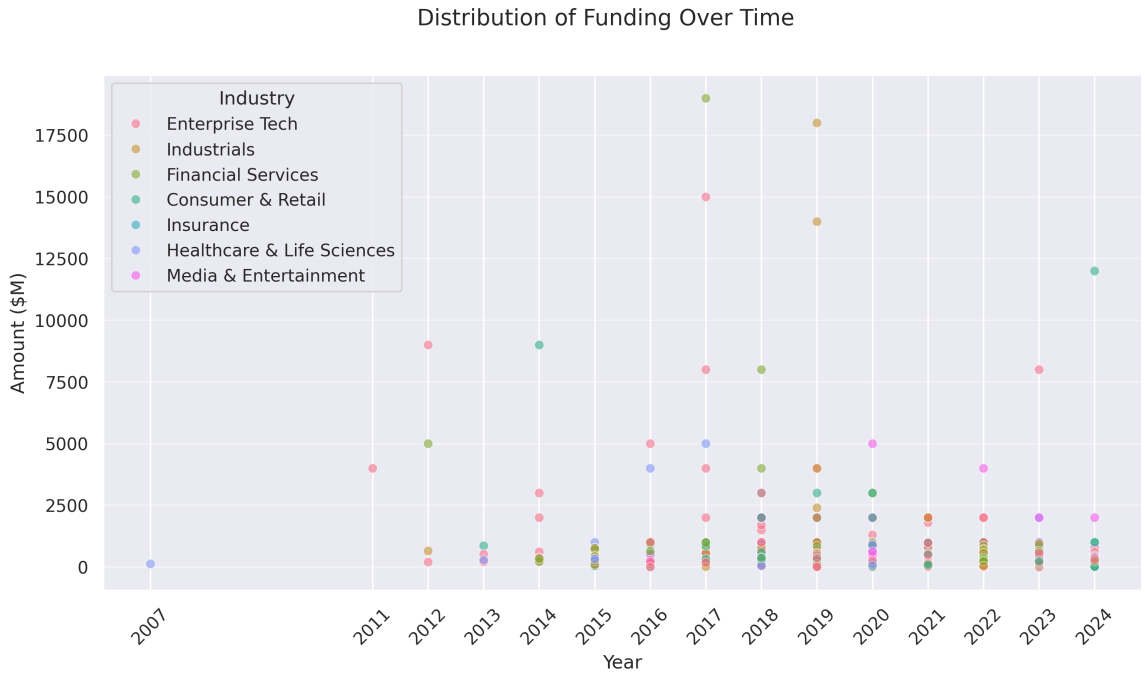
6.3 Distribution of Valuations Over Time

```
plt.subplots(figsize=(12, 6), dpi=300)
sns.scatterplot(df, x='Unicorn Year', y='Valuation ($B)', alpha=.6, hue='Industry')
plt.suptitle('Distribution of Valuations Over Time')
plt.xlabel('Year')
plt.ylabel('Amount ($B)')
plt.xticks(df['Unicorn Year'].unique(), rotation=45)
plt.grid(axis='y', alpha=0.5)
plt.yscale('log')
plt.show()
```



6.4 Distribution of Funding Over Time

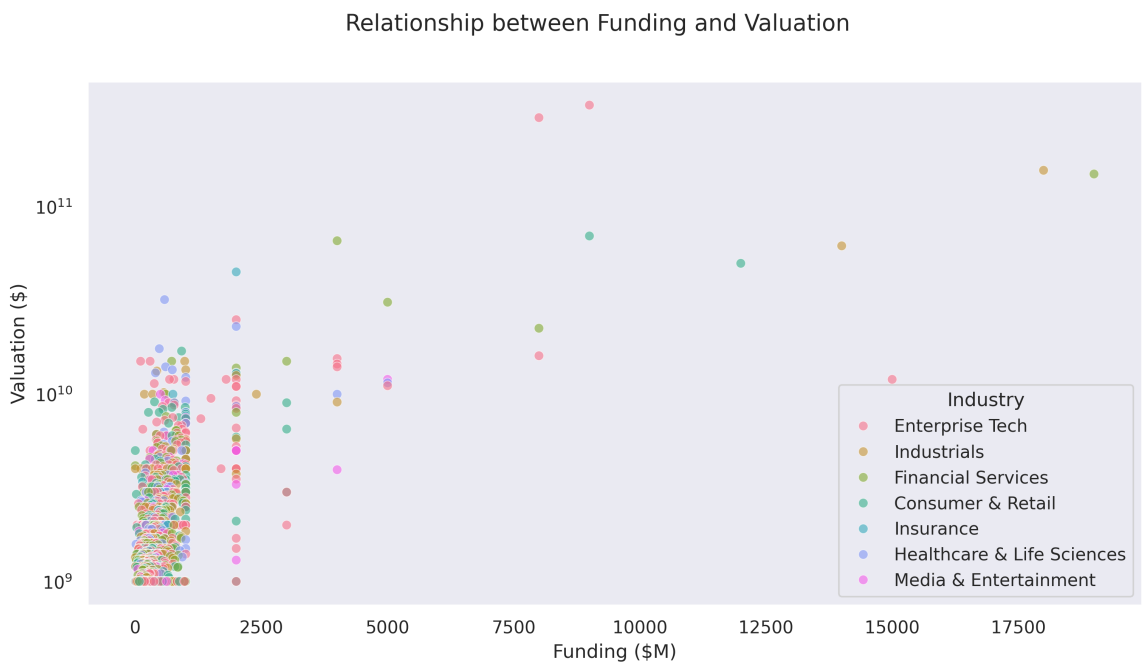
```
plt.subplots(figsize=(12, 6), dpi=300)
sns.scatterplot(df, x='Unicorn Year', y=df['Total Equity Funding ($)']/1e6, alpha=0.6,
↪  hue='Industry')
plt.suptitle('Distribution of Funding Over Time')
plt.xlabel('Year')
plt.ylabel('Amount ($M)')
plt.xticks(df['Unicorn Year'].unique(), rotation=45)
plt.grid(axis='y', alpha=0.5)
plt.show()
```



7 Correlation Analysis

7.1 Relationship between Funding and Valuation

```
plt.subplots(figsize=(12, 6), dpi=300)
sns.scatterplot(df, x=df['Total Equity Funding ($)']/1e6, y=df['Valuation ($B)']*1e9,
↪  alpha=0.6, hue='Industry')
plt.suptitle('Relationship between Funding and Valuation')
plt.xlabel('Funding ($M)')
plt.ylabel('Valuation ($)')
plt.grid()
# plt.xscale('log')
plt.yscale('log')
plt.show()
```



8 Historical Analysis

8.1 Survival and Acquisition

1. Find out companies no longer listed as unicorns in 2024

```
df_2022 = pd.read_csv('input/datasets/Unicorn_Companies (March 2022).csv')
df_2022['Valuation ($B)'] = pd.to_numeric(df_2022['Valuation ($B)'].str.replace('$',
↪  ''))
df_out = df_2022[~df_2022['Company'].str.lower().isin(df['Company'].str.lower())]
```

```
178 companies no longer listed in 2024 unicorn list
```

```
df_out.head()
```

2. Financial Stage

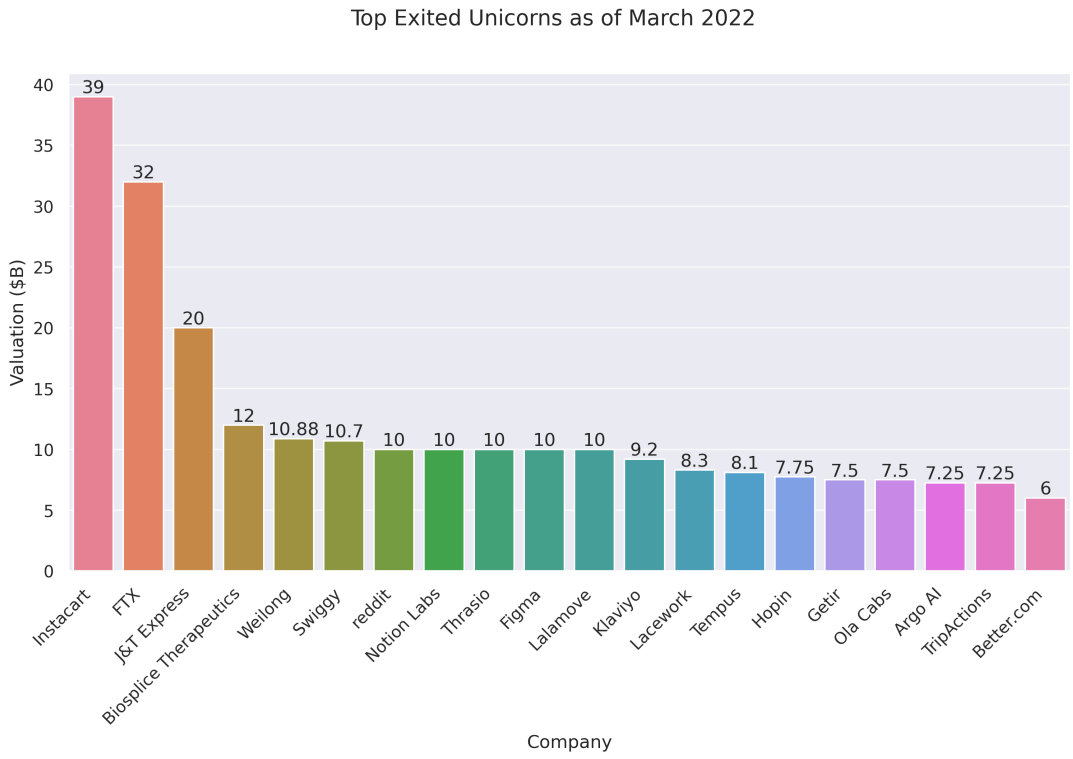
```
df_out.size()
```

```
Financial Stage
Acq             1
Acquired        7
Divestiture     1
IPO             2
dtype: int64
```

3. Top Exited Unicorns as of March 2022

```
df_top_companies = df_out.sort_values('Valuation ($B)', ascending=False).head(20)
df_top_companies
```

```
plt.subplots(figsize=(12, 6), dpi=300)
ax = sns.barplot(df_top_companies,
                 x='Company',
                 y='Valuation ($B)',
                 hue='Company')
for i in ax.containers:
    ax.bar_label(i)
plt.suptitle('Top Exited Unicorns as of March 2022')
plt.ylabel('Valuation ($B)')
plt.xlabel('Company')
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', alpha=0.75)
plt.show()
```



9 Funded by Y-Combinator

Y Combinator, founded in 2005 by Paul Graham and others, is a prestigious startup accelerator based in Silicon Valley that provides early-stage companies with seed funding, mentorship, and resources over a three-month program held twice a year. Startups receive initial funding in exchange for equity and culminate in a Demo Day where they

pitch to investors. Y Combinator has launched successful companies like Airbnb, Dropbox, and Stripe, significantly impacting the startup ecosystem and inspiring numerous other accelerators globally.

- **Datasets**
  - **YC Campanies**

```
df_yc_companies = pd.read_csv('input/datasets/2024 YCombinator All Companies
↪  Dataset/companies.csv')

df_yc_industries = pd.read_csv('input/datasets/2024 YCombinator All Companies
↪  Dataset/industries.csv')
df_yc_tags = pd.read_csv('input/datasets/2024 YCombinator All Companies
↪  Dataset/tags.csv')
# print(df_yc_tags.groupby('id')['tag'].agg(list).reset_index())
df_yc_companies = df_yc_companies.merge(df_yc_industries[['id',
↪  'industry']].groupby('id')['industry'].agg(list).reset_index(), on='id',
↪  how='left')
df_yc_companies =
↪  df_yc_companies.merge(df_yc_tags.groupby('id')['tag'].agg(list).reset_index(),
↪  on='id', how='left')
df_yc_companies = df_yc_companies[['name', 'slug', 'oneLiner', 'website',
↪  'smallLogoUrl', 'teamSize', 'tag', 'industry', 'batch']].rename(columns={
    'name': 'Company',
    'slug': 'Slug',
    'oneLiner': 'Short Description',
    'website': 'Website',
    'smallLogoUrl': 'Logo',
    'teamSize': 'Team Size',
    'tag': 'Tags',
    'industry': 'Industries',
    'batch': 'Batch'
})
print(df_yc_companies.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4844 entries, 0 to 4843
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Company            4844 non-null   object
 1   Slug               4841 non-null   object
 2   Short Description  4692 non-null   object
 3   Website            4817 non-null   object
 4   Logo               4197 non-null   object
 5   Team Size          4766 non-null   float64
 6   Tags               4463 non-null   object
 7   Industries         4825 non-null   object
```

```
 8   Batch              4844 non-null   object
dtypes: float64(1), object(8)
memory usage: 340.7+ KB
None
```

```
df2_yc_companies = pd.read_json('input/datasets/yc_startups.json')
print(df2_yc_companies.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   name         1000 non-null   object
 1   description  1000 non-null   object
 2   location     1000 non-null   object
 3   url          1000 non-null   object
 4   tags         1000 non-null   object
 5   site_url     999 non-null    object
 6   tag_line     999 non-null    object
 7   long_desc    999 non-null    object
 8   thumbnail    975 non-null    object
 9   founders     999 non-null    object
 10  meta         999 non-null    object
 11  socials      999 non-null    object
dtypes: object(12)
memory usage: 93.9+ KB
None
```

- **YC Founders**

```
df_yc_founders = pd.read_csv('input/datasets/2024 YCombinator All Companies
↪  Dataset/founders.csv')
print(df_yc_founders.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8465 entries, 0 to 8464
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   first_name       8461 non-null   object
 1   last_name        8456 non-null   object
 2   hnid             8465 non-null   object
 3   avatar_thumb     8465 non-null   object
 4   current_company  7624 non-null   object
```

```
 5   current_title    2201 non-null   object
 6   company_slug     8465 non-null   object
 7   top_company      8465 non-null   bool
dtypes: bool(1), object(7)
memory usage: 471.3+ KB
None
```

9.1 How many YC companies are in unicorn status currently?

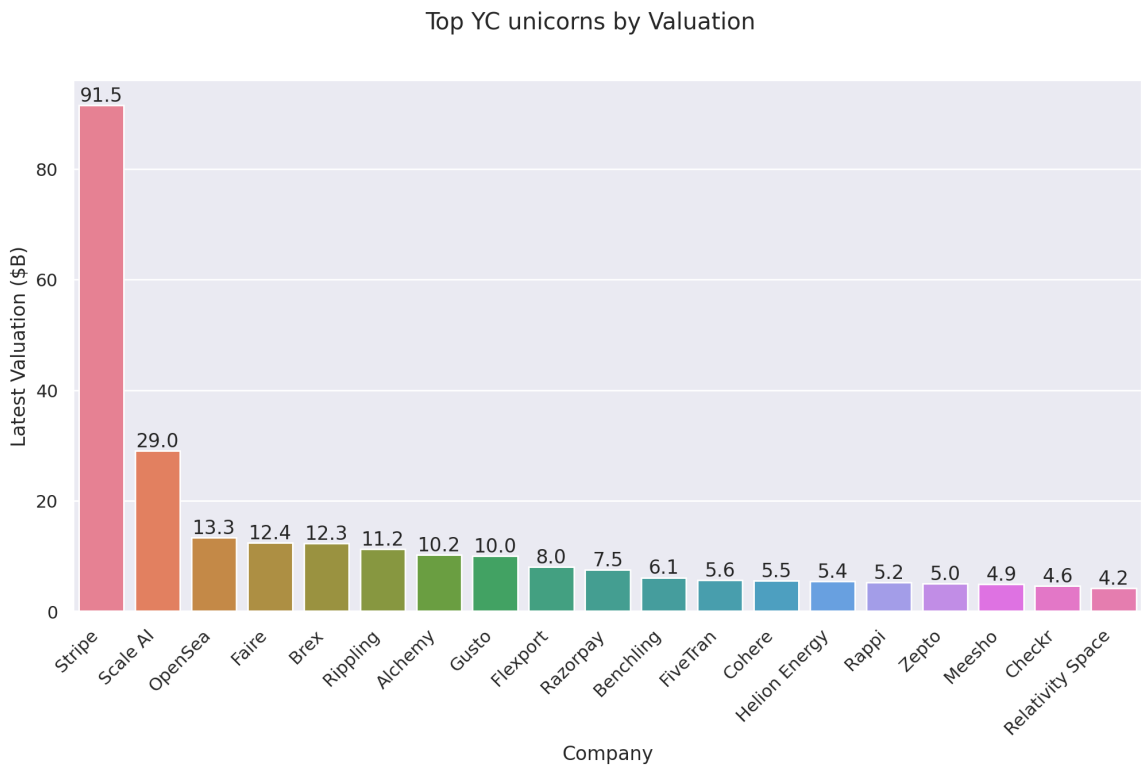
```
df_yc_unicorns = df.assign(tmp_col=df.Company.str.lower()).merge(
    df_yc_companies[['Company', 'Slug', 'Short Description', 'Website', 'Logo', 'Team
    ↪  Size', 'Tags', 'Industries', 'Batch']].assign(tmp_col=lambda x:
    ↪  x.Company.str.lower()),
    on='tmp_col', how='inner').drop(['tmp_col', 'Company_y'],
    ↪  axis=1).rename(columns={'Company_x': 'Company'})
df_yc_unicorns['Batch Season'] = df_yc_unicorns['Batch'].apply(lambda x: 'Summer' if
↪  x[0]=='S' else 'Winter')
df_yc_unicorns['Batch Year'] = pd.to_numeric(df_yc_unicorns['Batch'].apply(lambda x:
↪  f'20{x[1:]}'))
print(df_yc_unicorns.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 98 entries, 0 to 97
Data columns (total 24 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Company                    98 non-null     object
 1   Valuation ($B)             98 non-null     float64
 2   Total Equity Funding ($)   98 non-null     int64
 3   Unicorn Date               98 non-null     datetime64[ns]
 4   Date Founded               98 non-null     int64
 5   Years to Unicorn           98 non-null     object
 6   Industry                   98 non-null     object
 7   Country                    98 non-null     object
 8   City                       98 non-null     object
 9   Select Investors           98 non-null     object
 10  Unicorn Year               98 non-null     int32
 11  Funding ($B)               98 non-null     float64
 12  Latest Valuation ($B)      98 non-null     float64
 13  Years to Unicorn (Months)  98 non-null     int64
 14  Slug                       98 non-null     object
 15  Short Description          97 non-null     object
 16  Website                    98 non-null     object
 17  Logo                       95 non-null     object
```

```
 18  Team Size                  96 non-null     float64
 19  Tags                       92 non-null     object
 20  Industries                 98 non-null     object
 21  Batch                      98 non-null     object
 22  Batch Season               98 non-null     object
 23  Batch Year                 98 non-null     int64
dtypes: datetime64[ns](1), float64(4), int32(1), int64(4), object(14)
memory usage: 18.1+ KB
None
```

9.2 Top Companies by Valuation

```
df_top_yc_unicorns = df_yc_unicorns.sort_values(by='Latest Valuation ($B)',
↪  ascending=False).head(20)
fig, ax = plt.subplots(figsize=(12,6), dpi=200)
ax = sns.barplot(data=df_top_yc_unicorns, x='Company', y='Latest Valuation ($B)',
↪  hue='Company')
for i in ax.containers:
    ax.bar_label(i, fmt='%.1f')
plt.xticks(rotation=45, ha='right')
plt.suptitle('Top YC unicorns by Valuation')
plt.show()
```

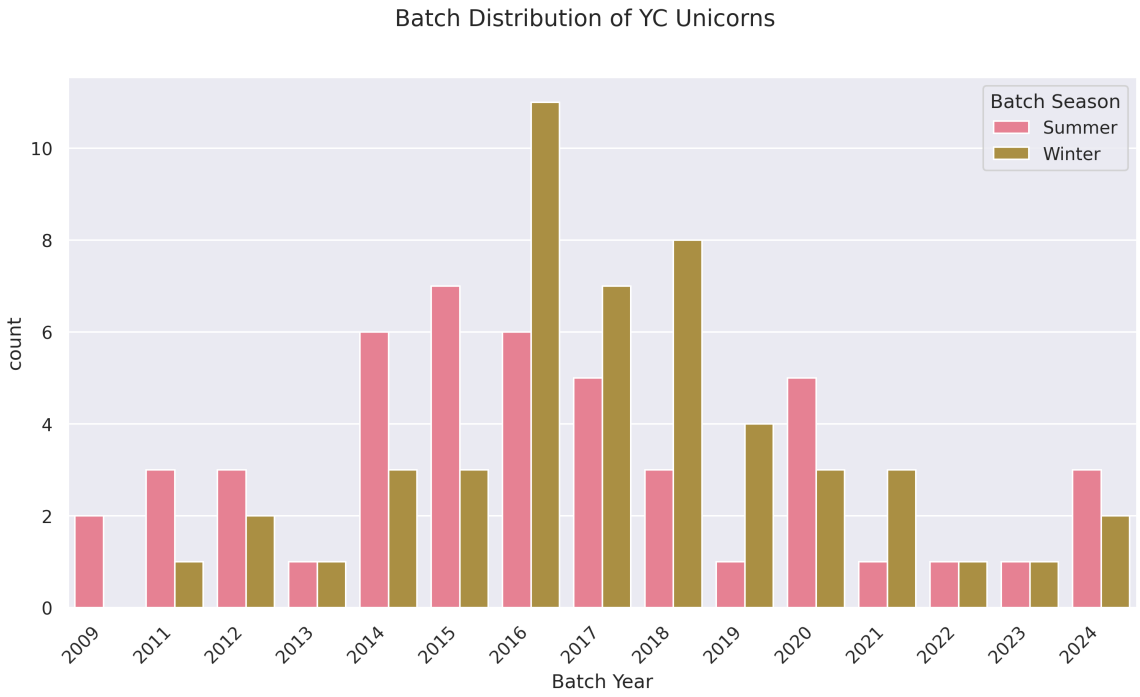


9.3 YC Batch Distribution

```
df_grouped = df_yc_unicorns.groupby(['Batch Year', 'Batch
↪  Season']).size().reset_index(name='count').sort_values(by='Batch Year')
print(df_grouped)
# sns.lmplot(df_group, x='Batch', y=')
```

	Batch Year	Batch Season	count
0	2009	Summer	2
1	2011	Summer	3
2	2011	Winter	1
3	2012	Summer	3
4	2012	Winter	2
5	2013	Summer	1
6	2013	Winter	1
7	2014	Summer	6
8	2014	Winter	3
9	2015	Summer	7
10	2015	Winter	3
11	2016	Summer	6
12	2016	Winter	11
14	2017	Winter	7
13	2017	Summer	5
15	2018	Summer	3
16	2018	Winter	8
17	2019	Summer	1
18	2019	Winter	4
19	2020	Summer	5
20	2020	Winter	3
21	2021	Summer	1
22	2021	Winter	3
23	2022	Summer	1
24	2022	Winter	1
25	2023	Summer	1
26	2023	Winter	1
27	2024	Summer	3
28	2024	Winter	2

```
plt.subplots(figsize=(12,6),dpi=300)
sns.barplot(df_grouped, x='Batch Year', y='count', hue='Batch Season')
plt.xticks(rotation=45, ha='right')
plt.suptitle('Batch Distribution of YC Unicorns')
plt.show()
```



9.4 Top Countires

```
top_countries = df_yc_unicorns['Country'].value_counts().nlargest(20).index
top_countries
```

```
Index(['United States', 'India', 'United Kingdom', 'Canada', 'Mexico', 'Indonesia', 'Co
```

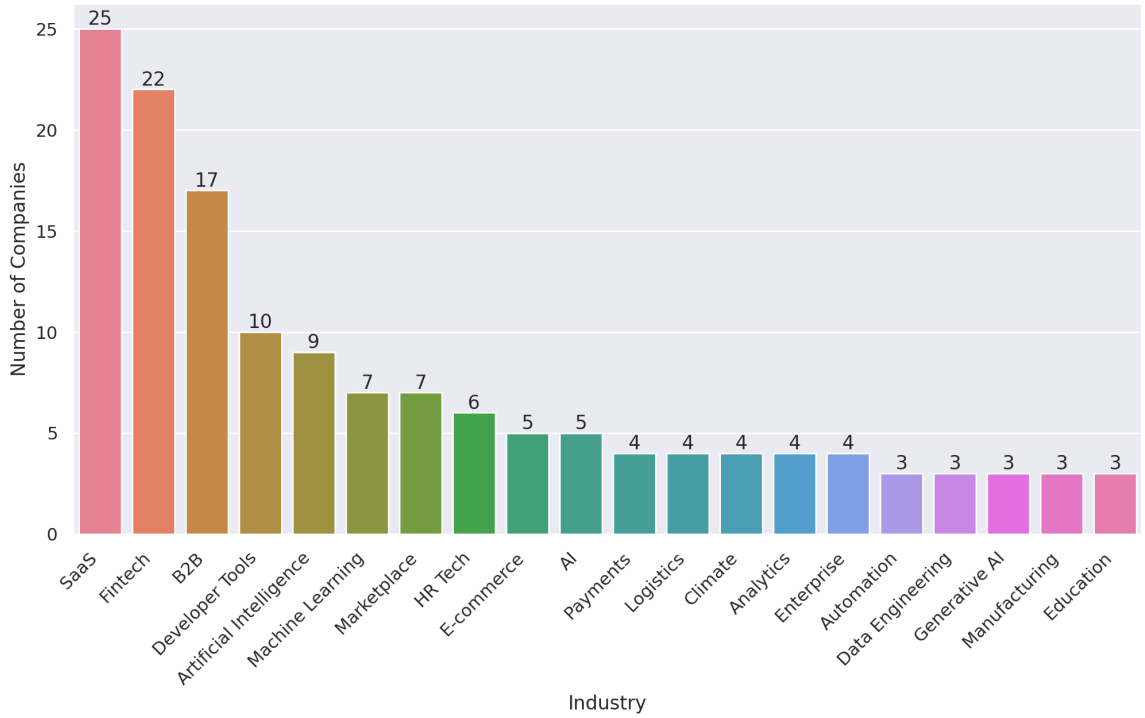
9.5 Top Industries

```
top_industries =
↪  df_yc_unicorns['Tags'].explode().value_counts().head(20).reset_index(name='Count')
print(top_industries)
```

```
                    Tags  Count
0                   SaaS     25
1                Fintech     22
2                    B2B     17
3         Developer Tools     10
4  Artificial Intelligence      9
5        Machine Learning      7
6             Marketplace      7
7                 HR Tech      6
8              E-commerce      5
9                      AI      5
```

```
10                 Payments      4
11                Logistics      4
12                  Climate      4
13                Analytics      4
14               Enterprise      4
15               Automation      3
16        Data Engineering       3
17           Generative AI       3
18           Manufacturing       3
19               Education       3
```

```
plt.subplots(figsize=(12,6), dpi=200)
ax = sns.barplot(data=top_industries, x='Tags', y='Count', hue='Tags')
ax.set(ylabel='Number of Companies',
       xlabel='Industry')
for i in ax.containers:
    ax.bar_label(i)
plt.xticks(rotation=45, ha='right')
plt.show()
```

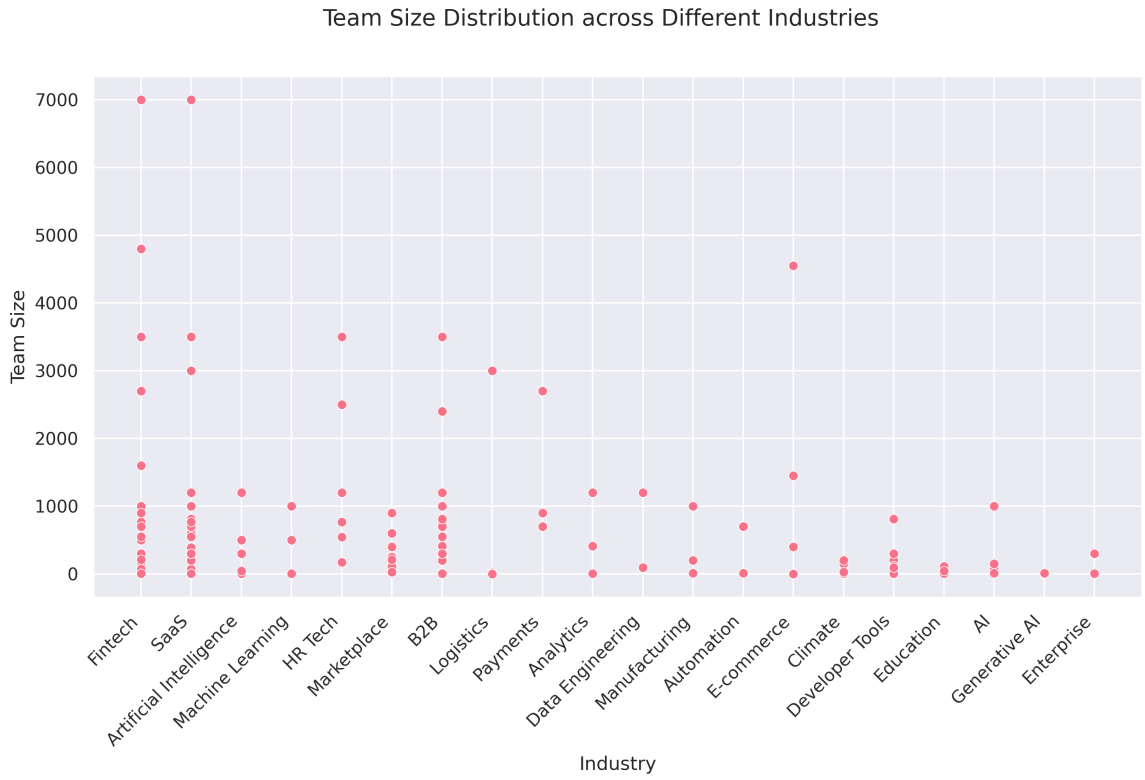


9.5.1 Team Size Distribution across Different Industries

```
df_scoped = df_yc_unicorns.explode('Tags')
df_scoped = df_scoped[df_scoped['Tags'].isin(top_industries['Tags'])]
df_scoped = df_scoped.dropna()

plt.subplots(figsize=(12,6), dpi=300)
```

```
ax = sns.scatterplot(df_scoped, x='Tags', y='Team Size')
ax.set(ylabel='Team Size',
       xlabel='Industry')
plt.xticks(rotation=45, ha='right')
plt.suptitle('Team Size Distribution across Different Industries')
plt.show()
```



10 References

- Unicorn (finance) [wikipedia]